DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

U23AIT53-DATA EXPLORATION AND VISULIZATION

QUESTION BANK

UNIT I CREATING AND MANIPULATING NUMPY ARRAYS 9

Creating arrays, indexing and slicing, mathematical operations with NumPy arrays

PART A

S. No	Question	BTL
1	**Define a NumPy array.** A NumPy array is a multi-dimensional data structure. It stores elements of the same data type in contiguous memory locations. It supports efficient mathematical and logical operations. It is the core data structure of the NumPy library.	BTL-1
2	**Write the command to import the NumPy library**. NumPy is usually imported with the alias np. The standard command used is import numpy as np. This makes it easier to call NumPy functions. Using the alias improves readability and reduces typing.	BTL-1
3	**Differentiate between Python list and NumPy array.** A Python list can store elements of different data types. A NumPy array can only store elements of the same type. Lists are slower for mathematical operations. NumPy arrays are faster and support vectorized operations	BTL-2
4	**Write a command to create a 1D array of numbers from 1 to 10.** NumPy provides the function arange() to create sequences. The command is np.arange(1,11). This generates numbers from 1 to 10. The last value is exclusive, so we use 11 as the stop.	BTL-3
5	**How do you create a NumPy array of all zeros of size 5?** NumPy provides the zeros() function. The command np.zeros(5) creates an array of size 5. All elements will be 0. By default, the data type is float.	BTL-3
6	**Give the syntax to create an array filled with ones.** NumPy provides the ones() function. The syntax is np.ones(shape). The shape can be a single number or a tuple. By default, the elements are of float type.	BTL-1
7	**Write a command to create an array of evenly spaced numbers between 0 and 1 with 5 values.** NumPy provides the linspace() function. The command np.linspace(0,1,5) generates 5 values. The values are evenly spaced between 0 and 1. Both endpoints are included.	BTL-3
8	What is the difference between linspace() and arange() functions? he linspace() function generates a fixed number of evenly spaced values.	BTL-2

	The arange() function generates values based on a fixed step size. In linspace(), we specify how many values to create. In arange(), we specify the step difference between values.	
9	Define array slicing with an example. Array slicing is extracting a part of an array. It uses the notation [start:end:step]. For example, arr[1:4] extracts elements from index 1 to 3. Slicing does not copy data but creates a view.	BTL-1
10	Write a command to access the third element of array arr = np.array([2,4,6,8,10]). NumPy arrays use zero-based indexing. The third element is at index 2. The command is arr[2]. The result will be 6.	BTL-3
11	How do you access the last two elements of an array using slicing? Negative indexing can be used to access elements from the end. The last two elements are obtained using arr[-2:]. This selects the second last element and the last element. It works for both 1D and 2D arrays.	BTL-3
12	Write a command to reverse a NumPy array. A NumPy array can be reversed using slicing. The syntax is arr[::-1]. This reverses the order of elements. It returns a new reversed array.	BTL-3
13	How do you reshape a 1D array of size 9 into a 3×3 matrix? NumPy provides the reshape() function. A 1D array of size 9 can be converted into 3 rows and 3 columns. The command is arr.reshape(3,3). The number of elements must remain the same.	BTL-3
14	Differentiate between reshape() and resize() functions. The reshape() function returns a new view of the array with changed dimensions. It does not modify the original array. The resize() function changes the shape of the original array. It may pad or truncate elements if needed.	BTL-2
15	Write a NumPy command to find the sum of all elements in an array. NumPy provides the function np.sum(). The command np.sum(arr) calculates the sum. It adds all elements of the array. We can also specify axis for multi-dimensional arrays.	BTL-3
16	Write the syntax to perform element-wise multiplication of two arrays. Element-wise multiplication is supported using the * operator. The syntax is arr1 * arr2. Both arrays must have the same shape or be broadcastable. Each element of the first array multiplies with the corresponding element of the second.	BTL-1
17	What is broadcasting in NumPy? Give an example. Broadcasting allows NumPy to operate on arrays of different shapes. A smaller array is broadcasted to match the shape of a larger one. For example, [1,2,3] + 5 gives [6,7,8]. This avoids writing explicit loops.	BTL-2
18	Write a command to calculate the mean of array elements. NumPy provides the function np.mean(). The command np.mean(arr) calculates the average value. It divides the sum of all elements by the number of elements. Axis can also be specified for multi-dimensional arrays.	BTL-3
19	How do you find the maximum and minimum elements of an array?	BTL-3

	NumPy provides np.max() and np.min() functions. The maximum value can be found using np.max(arr). The minimum value can be found using np.min(arr). Both can also be applied along specific axes.	
20	Write a NumPy function to generate an identity matrix of size 4. An identity matrix is a square matrix with ones on the diagonal. NumPy provides the identity() function. The command np.identity(4) generates a 4×4 identity matrix. The diagonal elements will be 1 and others will be 0.	BTL-3
21	Define vectorized operation in NumPy. A vectorized operation applies directly to an entire array without loops. For example, arr1 + arr2 adds corresponding elements. This makes computation faster. It is a key feature of NumPy.	BTL-1
22	Write a command to get the transpose of a 2D array. The transpose of an array is obtained using the .T attribute. The command is arr.T. It interchanges rows and columns. It is commonly used in matrix operations.	BTL-3
23	How do you perform matrix multiplication in NumPy? NumPy provides the dot() function. The command is np.dot(arr1, arr2) or arr1 @ arr2. This performs matrix multiplication, not element-wise multiplication. The number of columns in the first must match the number of rows in the second.	BTL-3
24	Write a command to flatten a 2D array into 1D. A 2D array can be converted into 1D using flatten(). The command is arr.flatten(). This returns a copy of the array as a 1D sequence. The original array remains unchanged.	BTL-3
25	Define the difference between shallow copy and deep copy in NumPy arrays. A shallow copy creates a new array object but shares the same data with the original. Changes in one affect the other. A deep copy creates a completely new array with its own data. Changes in one do not affect the other.	BTL-2

PART B

S. No	Question	BTL
1	Explain various ways to create NumPy arrays with suitable examples.	BTL-2
2	Draft a Python NumPy program to depict indexing in NumPy arrays.	BTL-3
3	Develop a NumPy program to illustrate slicing operations on NumPy arrays.	BTL-3
6	Construct a program in Python using NumPy to show element-wise arithmetic operations on arrays.	BTL-3
7	Implement a Python program with NumPy to explain mathematical functions.	BTL-3
9	Explain broadcasting in NumPy with a program.	BTL-2
10	Write a program to flatten a 2D array into 1D using flatten() and ravel().	BTL-3

S. No	Question	BTL
11	Write a Python program using NumPy to create the following arrays: (a) A 1D array from 0 to 20 with step size 2 (b) A 3x3 identity matrix (c) A 4x4 matrix with random integers between 10 and 50 (d) A 5x5 zero matrix	BTL-3 (Apply)
12	Write a Python program to demonstrate **indexing and slicing**: (a) Create an array `[10,20,30,40,50,60,70,80]` (b) Extract elements from index 2 to 5 (c) Print alternate elements (d) Reverse the array	BTL-3 (Apply)
13	Create two arrays `A = [2,4,6,8]` and `B = [1,3,5,7]`. Write a program to perform: (i) Addition (ii) Subtraction (iii) Multiplication (iv) Division (v) Exponentiation	BTL-3
14	Write a Python program to: (a) Create a 1D array of numbers 1 to 12 (b) Reshape it into 3x4 and 2x6 arrays (c) Flatten and Ravel the reshaped array (d) Transpose the reshaped array	BTL-3
15	Create an array of integers from 1 to 30. Write a program to: (i) Extract even numbers (ii) Extract odd numbers (iii) Extract numbers divisible by 5 (iv) Replace all odd numbers with `-1`	BTL-3

UNIT II DATA MANIPULATION WITH PANDAS

Series and Data Frame objects; importing and exporting data from various file formats into pandas Data Frame; Data selection and filtering indexing, slicing, conditional filtering using boolean indexing

PART A

S. No	Question	BTL
1.	**Define a Pandas Series.** (i) A Pandas Series is a one-dimensional labeled array. (ii) It can hold any data type (int, float, string, etc.). (iii) Labels are called **index**. (iv) Created using `pd.Series()`.	BTL-1
2.	**Define a Pandas DataFrame.**) A DataFrame is a 2D tabular data structure.	BTL-1

S. No	Question	BTL
	(ii) It consists of rows and columns. (iii) Each column is like a Series. (iv) Created using `pd.DataFrame()`	
3.	**Write the command to create a Series from a list `[10, 20, 30]`.** import pandas as pd s = pd.Series([10, 20, 30]) print(s)	BTL-3

4.	**Write the command to create a DataFrame from a dictionary.**
	import pandas as pd
	data = {
	"Name": ["Alice", "Bob", "Charlie"],
	"Age": [24, 27, 22],
	"City": ["New York", "Paris", "London"]
	}
	df = pd.DataFrame(data)
	print(df)
		BTL-3

5.	**How to read a CSV file into a DataFrame?** (i) Import pandas. (ii) Use `pd.read_csv("filename.csv")`. (iii) Reads data into DataFrame. (iv) Returns tabular data with columns.	BTL-1
6.	**Write the command to export a DataFrame to Excel.** (i) Import pandas. (ii) Create DataFrame. (iii) Use `df.to_excel("output.xlsx")`. (iv) Saves DataFrame as Excel file.	BTL-3
7.	**Mention two file formats supported by Pandas.** (i) CSV (Comma-Separated Values). (ii) Excel (.xlsx). (iii) JSON. (iv) SQL databases.	BTL-1
8.	**Differentiate between `read_csv()` and `read_excel()`.** `read_csv()` reads CSV files. (ii) `read_excel()` reads Excel sheets. (iii) `read_csv()` faster for plain text. (iv) `read_excel()` supports multiple sheets	BTL-2

9.	data = {
	"Name": ["Alice", "Bob", "Charlie", "David", "Eva", "Frank"],
	"Age": [24, 27, 22, 32, 29, 30],
	"City": ["New York", "Paris", "London", "Tokyo", "Berlin", "Sydney"]
	}
	df = pd.DataFrame(data)
	Write code to display the first 5 rows of a DataFrame.
	# Display the first 5 rows
	print(df.head())
	Write code to display the last 3 rows of a DataFrame.
	print(df.tail(3))
	Write code to select the first row using `iloc`.
	first_row = df.iloc[0]
	Write code to select rows 0 to 2 and columns "Name" to "Age".
	subset = df.loc[0:2, "Name":"Age"]
	Write code to select alternate rows of a DataFrame.
		BTL-3

S. No	Question	BTL

	alternate_rows = df.iloc[::2] # step of 2, starts from 0
	print("Alternate Rows:\n", alternate_rows)
	Write code to filter rows where Age > 25.
	filtered_df = df[df["Age"] > 25]
	print("Rows where Age > 25:\n", filtered_df)
	Write code to filter rows where City = "London".
	# Filter rows where City = "London"
	filtered_df = df[df["City"] == "London"]
	print("Rows where City = 'London':\n", filtered_df)
	Write code to filter rows where Age > 25 and City = "Paris".
	filtered_df = df[(df["Age"] > 25) & (df["City"] == "Paris")]
	print("Rows where Age > 25 and City = 'Paris':\n", filtered_df)	

10.	**How do you access a single column in a DataFrame?** Use a list of column names. 2. Syntax: `df[['col1','col2']]`. 3. It returns a smaller DataFrame. 4. Useful for analyzing specific data	BTL-1
11.	**Differentiate between `iloc` and `loc`.** **Aspect** / **iloc** / **loc** **Meaning**: Integer-location based indexing / Label-based indexing **Index Type**: Uses **row/column positions** (0,1,2,…) / Uses **row/column labels** (names/indices) **Example**: `df.iloc[0, 1]` → 1st row, 2nd column / `df.loc[0, "Age"]` → Row with index 0, column "Age" **Use Case**: When positions are known but labels may change / When labels are known for precise selection	BTL-2
12.	**What is slicing in Pandas?** **Definition** – Slicing in Pandas refers to selecting a specific portion (subset) of rows or columns from a DataFrame or Series. **Row slicing** – It allows you to extract a range of rows using index labels (`loc`) or integer positions (`iloc`). **Column slicing** – Columns can also be sliced by specifying a range of column labels or positions.	BTL-1
13.	**Define Boolean indexing**. Boolean indexing in Pandas is a powerful technique used to filter data based on conditions. When a condition is applied to a DataFrame or Series, it generates a Boolean mask consisting of `True` and `False` values for each element. Only the rows where the condition evaluates to `True` are selected, while others are excluded	BTL-1
14.	**How do you check for missing values in a DataFrame?** In **Pandas**, missing values are usually represented as `NaN` (Not a Number). • **`df.isnull()`** → Returns `True` for missing values and `False` otherwise. • **`df.isnull().sum()`** → Gives the total count of missing values in each column. • **`df.isna()`** → Same as `isnull()`, used to detect missing values. • **`df.info()`** → Displays non-null counts, which helps to identify missing values.	BTL-1
15.	**Write a command to drop rows with missing values.** **df = df.dropna()**	BTL-3

S. No	Question	BTL

16.	import pandas as pd
	import numpy as np
	
	# Sample DataFrame with missing values
	data = {
	"Name": ["Alice", "Bob", "Charlie", None],
	"Age": [24, np.nan, 22, 32],
	"City": ["New York", "Paris", None, "Tokyo"]
	}
	df = pd.DataFrame(data)
	print("Original DataFrame:\n", df)
	# Drop rows with missing values
	df = df.dropna()
	print("\nDataFrame after dropping rows with missing values:\n", df)
	Write a command to replace missing values with 0.
	df = df.fillna(0)
	import pandas as pd
	import numpy as np
	
	# Sample DataFrame with missing values
	data = {
	"Name": ["Alice", "Bob", "Charlie", None],
	"Age": [24, np.nan, 22, 32],
	"City": ["New York", "Paris", None, "Tokyo"]
	}
	df = pd.DataFrame(data)
	print("Original DataFrame:\n", df)
	
	# Replace missing values with 0
	df = df.fillna(0)
	print("\nDataFrame after replacing missing values with 0:\n", df)
		BTL-3

PART B

S. No	Question	BTL Level
1	Explain the difference between Pandas Series and DataFrame with suitable examples. Write a program to create both and display their values.	BTL-2, BTL-3
2	Discuss the importance of importing and exporting data in Pandas. Write a Python program to import data from a CSV file into a DataFrame and export the DataFrame into Excel format.	BTL-2, BTL-3
3	Define indexing and slicing in Pandas. Write a program to select the first 3 rows and 2 columns of a DataFrame and also select alternate rows.	BTL-1, BTL-3
4	Differentiate between loc and iloc in Pandas. Write a program to demonstrate selection of rows and columns using both.	BTL-2, BTL-3
5	Explain conditional filtering in Pandas. Write a program to filter rows where Age > 25, City = "Paris", and Age > 25 and City = "Paris".	BTL-2, BTL-3
6	What are missing values in Pandas? Explain different methods to handle them. Write a program to check missing values, replace missing values with 0, and drop rows with missing values.	BTL-1, BTL-2, BTL-3
7	Describe the process of data selection in Pandas. Write a Python program to select a single column, multiple columns, and rows using slicing.	BTL-2, BTL-3
8	What are the different file formats supported by Pandas for importing and	BTL-1, BTL-

S. No	Question	BTL Level
	exporting data? Write a program to read from a CSV file and write to a JSON file.	3
9	Define Boolean Indexing in Pandas with an example. Write a Python program to filter employee details where Salary > 50000 and Department = "IT".	BTL-1, BTL-3
10	Explain the concept of DataFrame modification. Write a program to add a new column "Total Marks", update marks of a student, and delete a column.	BTL-2, BTL-3

UNIT III GROUPING AND AGGREGATION WITH PANDAS

9

Grouping data using Pandas, applying aggregation functions such as sum, mean, count, etc. to grouped data, using pivot tables and cross-tabulation for data summarization

S. No	Question	BTL
1	Define grouping in Pandas. **Grouping in Pandas** refers to the process of splitting data into groups based on the values in one or more columns, so that we can apply operations like **aggregation, transformation, or filtering** on each group separately. □ It is mainly done using the **`groupby()`** function, which allows you to analyze subsets of data efficiently.	BTL-1
2	What is the use of `groupby()` in Pandas? The **`groupby()` function in Pandas** is used to **split data into groups** based on one or more columns and then perform operations like aggregation, transformation, or filtering on those groups. 1. `groupby()` helps in analyzing subsets of data by grouping rows that share a common value. 2. It is mainly used for applying **aggregate functions** like `sum()`, `mean()`, `count()`, etc. on grouped data. 3. It follows the **Split-Apply-Combine** strategy: split data → apply function → combine results. 4. Useful in data summarization, reporting, and statistical analysis.	BTL-1
3	Write the syntax of `groupby()` in Pandas. DataFrame.groupby(by=None, axis=0, level=None, as_index=True, sort=True, group_keys=True)	BTL-1
4	Differentiate between `groupby()` and `pivot_table()`. **groupby()**: Used for grouping data and applying aggregate functions, more flexible for transformations. **pivot_table()**: Creates a spreadsheet-like summary table with rows and columns, mainly for summarization.	BTL-2
5	Mention any two advantages of grouping in Pandas. 1. Simplifies data summarization by applying aggregate functions on groups. 2. Improves analysis by allowing comparisons between categories.	BTL-2
6	What is an aggregation function in Pandas? An aggregation function is a function that reduces multiple values of a group into a single value, e.g., `sum()`, `mean()`, `count()`.	BTL-1
7	Write the syntax of `agg()` function in Pandas. DataFrame.groupby('column').agg({'col1': 'sum', 'col2': 'mean'})	BTL-1

S. No	Question	BTL
8	List any four common aggregation functions. `sum()`, `mean()`, `count()`, `min()`, `max()`	BTL-1
9	Differentiate between `sum()` and `mean()` aggregation. **sum()**: Returns the total sum of values in each group. **mean()**: Returns the average of values in each group.	BTL-2
10	What does `count()` function return when applied on grouped data? It returns the number of non-null entries in each group.	BTL-1
13	What is the purpose of `size()` function in groupby? The `size()` function returns the total number of rows in each group, including NaN values.	BTL-1
14	Differentiate between `size()` and `count()`. **size()**: Counts all rows in a group (including NaN). **count()**: Counts only non-null values in each group.	BTL-2
15	Write the command to apply multiple aggregation functions (`sum`, `mean`) on grouped data. df.groupby('Department')['Salary'].agg(['sum', 'mean'])	BTL-3
16	Define pivot table in Pandas. A pivot table in Pandas is a data summarization tool that reshapes data into a table format, allowing aggregation of values based on rows and columns.	BTL-1
17	Write the syntax of `pivot_table()`. pd.pivot_table(data, values=None, index=None, columns=None, aggfunc='mean')	BTL-1
18	Differentiate between pivot table and cross-tabulation. **Pivot table**: Summarizes numerical data using aggregation functions like `mean`, `sum`. **Cross-tabulation**: Summarizes categorical data into frequency counts.	BTL-2
19	Write a command to create a pivot table showing average marks per subject.	BTL-3
20	Define cross-tabulation in Pandas. **Pivot table**: Summarizes numerical data using aggregation functions like `mean`, `sum`. **Cross-tabulation**: Summarizes categorical data into frequency counts.	BTL-1
21	Write the syntax of `pd.crosstab()`. Cross-tabulation (crosstab) is a method to compute a frequency table of two or more categorical variables.	BTL-1
22	Mention two use cases of cross-tabulation. Cross-tabulation (crosstab) is a method to compute a frequency table of two or more categorical variables.	BTL-2
24	Write a command to apply a custom aggregation function using `lambda`. df.groupby('Department')['Salary'].agg(lambda x: x.max() - x.min())	BTL-3
25	What is hierarchical (multi-level) grouping in Pandas? Hierarchical grouping means grouping data by more than one column, creating a multi-level index. Example: `df.groupby(['Department', 'Gender']).mean()`	BTL-1

PART B

S. No	Question	BTL
1	Explain the concept of grouping in Pandas with an example. Write a program to group employees based on their department and display the total salary for each department.	BTL-4
2	What is an aggregation function in Pandas? Discuss with examples.	BTL-3

S. No	Question	BTL
	Write a program to calculate the mean, maximum, and minimum sales of products grouped by region.	
3	Differentiate between `sum()`, `mean()`, and `count()` functions in grouped data. Write a program to group student records by subject and calculate the total, average, and count of marks.	BTL-2
4	Explain the use of `agg()` function in Pandas with an example. Write a program to group employee data by department and apply multiple aggregation functions (sum, mean) on their salaries.	BTL-4
5	Define a Pivot Table in Pandas. How is it useful in summarization? Write a program to create a pivot table showing the average marks scored by students in each subject.	BTL-3
6	Differentiate between `groupby()` and `pivot_table()` with suitable examples. Write a program to group data using both methods and compare the results.	BTL-2
7	What is cross-tabulation? Mention its uses. Write a program to create a cross-tabulation between Gender and Department of employees.	BTL-3
8	Explain hierarchical grouping in Pandas with an example. Write a program to group employee data by Department and Gender, and then calculate the mean salary for each group.	BTL-4
9	How does `size()` differ from `count()` in Pandas? Illustrate with an example. Write a program to demonstrate the difference using student data grouped by subject.	BTL-2
10	Write short notes on the Split–Apply–Combine strategy in Pandas. Write a program to group products by category and apply custom aggregation using a lambda function (e.g., range = max - min).	BTL-5

UNIT IV DATA VISUALIZATION WITH MATPLOTLIB AND SEABORN 9

Introduction to Matplotlib and Seaborn to plot data using figures and subplots, Plots - Line plots, scatter plots, and bar plots, visualizing distributions using histogram and box plots, Customizing plot aesthetics and adding annotations

PART A

Q.No	Question	BTL
1	Define Matplotlib. Matplotlib is a Python library used for data visualization. It helps in creating different types of plots such as line plots, bar plots, and scatter plots. It provides great flexibility to customize figures with titles, labels, colors, and legends. It is mainly used for 2D plots but also supports basic 3D plotting.	BTL-1
2	What is Seaborn? Seaborn is a statistical data visualization library built on top of Matplotlib. It comes with attractive default themes and color palettes for better presentation. Seaborn simplifies complex visualization tasks with minimal code. It is especially useful for plotting distributions, heatmaps, and categorical data	BTL-1
3	Write the command to import Matplotlib. Matplotlib can be imported using the command `import matplotlib.pyplot as plt`. The alias `plt` is commonly used to make function calls shorter and easier. Once imported, different plotting functions such as `plt.plot()` can be used. Finally, we can display the graph using `plt.show()`.	BTL-1

Q.No	Question	BTL
4	Write the command to import Seaborn. Seaborn is usually imported using `import seaborn as sns`. The alias `sns` is widely used by programmers for simplicity. After importing, visualization functions such as `sns.scatterplot()` or `sns.histplot()` can be used directly. This makes it efficient and easier to handle statistical plots.	BTL-1
5	Differentiate between Matplotlib and Seaborn. Matplotlib is a low-level library that requires more code but gives more customization options. Seaborn is a high-level library that provides easy-to-use functions with attractive styles. Matplotlib is often used for simple and customized plots, while Seaborn is used for statistical and complex visualizations. Both libraries are complementary and can be used together.	BTL-2
6	What is the use of `figure()` in Matplotlib? The `figure()` function is used to create a new figure window. It serves as a container for all plots, subplots, labels, and legends. Multiple figures can be created using this function in the same program. It helps to organize and manage multiple visualizations separately.	BTL-1
7	What is the use of `subplot()` in Matplotlib? The `subplot()` function allows multiple plots to be drawn in the same figure window. It divides the figure into a grid and places plots within specified cells. Each subplot can be indexed and customized individually. This function is very useful for comparing different plots side by side.	BTL-1
8	Define Line Plot? A line plot connects data points using straight lines. It is mainly used to represent trends and changes over time. The x-axis usually represents independent variables, while the y-axis represents dependent variables. Line plots are commonly used in time-series and trend analysis.	BTL-1
9	Write the syntax of a line plot. The syntax of a line plot is `plt.plot(x, y)`. Here, `x` and `y` are the datasets representing the axes. We can add labels, markers, and colors for customization. Finally, `plt.show()` is used to display the line plot.	BTL-1
10	Define a scatter plot. A scatter plot uses points to show the relationship between two variables. Each point in the plot represents an observation from the dataset. The x and y coordinates define the location of the point. Scatter plots are useful for identifying patterns, correlations, and outliers.	BTL-1
11	Write the syntax of a scatter plot. The syntax in Matplotlib is `plt.scatter(x, y)`. In this, `x` and `y` represent the data points. In Seaborn, we can use `sns.scatterplot(x, y, data=df)` for a simpler implementation. Scatter plots can also be customized with color, size, and shape of points.	BTL-1
12	Define a bar plot. A bar plot uses rectangular bars to represent data values. The length or height of each bar corresponds to the value it represents. Bar plots are commonly used for comparing categories. They can be displayed either vertically or horizontally.	BTL-1
13	Write the syntax of a bar plot. In Matplotlib, the syntax is `plt.bar(x, height)`. Here, `x` represents categories, and `height` represents their values. In Seaborn, we can use `sns.barplot(x, y, data=df)` to create bar plots. These plots are useful for comparing categorical data visually.	BTL-1
14	Differentiate between line plot and scatter plot.	BTL-2

Q.No	Question	BTL
	A line plot connects data points with lines, while a scatter plot uses only points. Line plots are better suited for showing trends over continuous data. Scatter plots are mainly used to show relationships and correlations between variables. Both plots serve different purposes in data visualization	
15	What is a histogram? A histogram shows the frequency distribution of numerical data. The data is divided into intervals known as bins. The height of each bar represents the frequency of values within that bin. Histograms are useful for understanding the shape and spread of data.	BTL-1
16	Write the syntax of a histogram. In Matplotlib, the syntax is `plt.hist(data, bins=10)`. Here, `data` is the dataset, and `bins` specify the number of intervals. In Seaborn, we can use `sns.histplot(data)`. Both functions are useful for showing the distribution of data	BTL-1
17	What is a box plot? A box plot is a graphical representation of the distribution of data. It shows minimum, first quartile, median, third quartile, and maximum values. Outliers are displayed as separate points outside the whiskers. Box plots are useful for comparing distributions across multiple groups	BTL-1
18	Write the syntax of a box plot. In Matplotlib, the syntax is `plt.boxplot(data)`. This creates a box plot for the dataset provided. In Seaborn, the syntax is `sns.boxplot(x, y, data=df)`. Box plots provide a summary of the spread and skewness in the data.	BTL-1
19	Differentiate between histogram and box plot. A histogram uses bins to show the frequency distribution of data. A box plot shows quartiles and outliers to summarize the spread of data. Histograms are better for visualizing the overall shape of a distribution. Box plots are better for comparing medians and detecting outliers.	BTL-2
20	What are plot aesthetics in Seaborn? Plot aesthetics in Seaborn refer to the visual styles applied to plots. They include themes, colors, and scaling options for better readability. Aesthetics improve the presentation of graphs and make them more informative. Seaborn provides built-in functions to control these visual styles.	BTL-1
21	Mention two functions used to customize aesthetics in Seaborn. The function `sns.set_style()` is used to adjust the background style of the plot. The function `sns.set_context()` changes the scaling of plot elements depending on the use case. These functions help in improving the readability and visual quality of the plot. They are especially useful when preparing plots for presentations.	BTL-1
22	Write the command to add a title in Matplotlib. The command is `plt.title("Graph Title")`. This adds a title above the graph. A title helps viewers understand the purpose of the graph quickly. It improves the clarity and readability of the visualization.	BTL-1
23	Write the command to label x and y axes in Matplotlib. The commands are `plt.xlabel("X-axis")` and `plt.ylabel("Y-axis")`. These labels specify what the axes represent. Without axis labels, the graph may look incomplete or confusing. Axis labels improve the interpretability of a plot.	BTL-1
24	What is the purpose of annotations in plots? Annotations are used to highlight specific points in a plot. They provide additional explanations or comments within the graph. Annotations make plots more informative and suitable for reports. They are useful when we want to emphasize	BTL-1

Q.No	Question	BTL
	important findings in the visualization.	
25	Write the command to add an annotation in Matplotlib. The command is `plt.annotate("Label", xy=(x, y), xytext=(x+1, y+1), arrowprops=dict(arrowstyle="->"))`. This adds a label at the specified point with an arrow. It is used to highlight important data points. Annotations make plots clearer and more meaningful for viewers.	BTL-1

PART B

S.No	Question	BTL
1	Write a Python program using **Matplotlib** to create a figure with two subplots: one showing a line plot of monthly sales and another showing a bar plot of the same data.	BTL-3
2	Explain the difference between **Matplotlib** and **Seaborn** with suitable examples. Write a Python program using Seaborn to draw a scatter plot of marks obtained by students in two subjects.	BTL-4
3	Write a program to demonstrate a **line plot** of years (x-axis) vs population growth (y-axis) using Matplotlib. Add appropriate title, x-axis label, and y-axis label.	BTL-3
4	Using Seaborn, write a program to create a **scatter plot** for "age vs salary" of employees and highlight gender using different colors. Explain the usefulness of scatter plots.	BTL-4
5	Write a program to draw a **bar plot** using Matplotlib to represent sales of different products. Add labels to axes, a title, and display values above bars.	BTL-3
6	What is a histogram? Write a Python program using Seaborn to visualize the **distribution of student marks** using a histogram.	BTL-2
7	Write a program to plot a **box plot** using Seaborn to compare salary distribution of employees in different departments. Explain how outliers are shown in a box plot.	BTL-4
8	Write a Python program to **customize plot aesthetics** using Seaborn's `set_style()` and `set_context()` functions. Draw a line plot to demonstrate the changes.	BTL-3
9	Write a program to demonstrate the use of **annotations** in Matplotlib by plotting a line graph of stock prices and marking the highest price point with an annotation.	BTL-3
10	Compare **histogram and box plot** with suitable examples. Write Python code to plot both for the same dataset and explain the differences in interpretation.	BTL-4

UNIT V NOSQLDATA MANAGEMENT FOR BIG DATA AND VISUALIZATION 9

Introduction to Plotly library for interactive visualization; Creating interactive line plots, scatter plots, and bar plots; Adding interactivity with hover effects, zooming, and panning

PART A

S.No	Question & Answer	BTL
1	**What is Plotly in Python?** Plotly is an open-source graphing library in Python used for creating interactive and dynamic visualizations. It supports line, bar, scatter, histogram, box plots, and advanced 3D visualizations. It is widely used in data science and analytics.	BTL-1
2	**Mention two advantages of using Plotly.** Plotly provides highly interactive charts with zooming, hovering, and panning features. It also integrates easily with Jupyter Notebook and dashboards, making it	BTL-1

S.No	Question & Answer	BTL
	user-friendly for analysis.	
3	**Write the command to install Plotly.** The command is `pip install plotly`. This installs the latest version of Plotly library in Python.	BTL-1
4	**How to import Plotly in Python?** We can import Plotly using `import plotly.express as px` or `import plotly.graph_objects as go`. Both are commonly used modules for creating charts.	BTL-1
5	**What is the difference between Matplotlib and Plotly?** Matplotlib produces static plots while Plotly creates interactive and dynamic plots. Plotly provides built-in interactivity like hover tooltips and zooming, which requires extra coding in Matplotlib.	BTL-2
6	**Define interactive visualization.** Interactive visualization allows users to interact with data through zooming, panning, and hovering. It makes data exploration easier and more meaningful.	BTL-1
7	**Give an example of an interactive feature in Plotly.** Hover effect is a common feature in Plotly. When the cursor moves over a data point, it displays values and labels dynamically.	BTL-1
8	**Write a simple command to create a line plot in Plotly Express.** Example: `px.line(df, x="Year", y="Sales", title="Yearly Sales")`. This generates an interactive line chart.	BTL-3
9	**Write the command to create a scatter plot in Plotly Express.** `px.scatter(df, x="Age", y="Salary", color="Gender")`. This produces a scatter plot with colors based on gender.	BTL-3
10	**How to create a bar plot in Plotly?** Example: `px.bar(df, x="Department", y="Employees")`. This creates an interactive bar chart with hover information.	BTL-3
11	**What is hover effect in Plotly?** Hover effect shows additional information when the mouse pointer is placed on a data point. It helps users read details without cluttering the chart.	BTL-1
12	**What is the use of zooming in Plotly?** Zooming allows focusing on a specific range of data in the chart. It is useful for analyzing large datasets in detail.	BTL-2
13	**What is panning in Plotly?** Panning allows users to drag and move across different parts of a plot. It helps navigate wide datasets effectively.	BTL-1
14	**Mention two real-time applications of Plotly.** Plotly is used in financial dashboards to visualize stock price changes. It is also used in research to analyze scientific datasets interactively.	BTL-2
15	**Write a command to display a histogram using Plotly.** Example: `px.histogram(df, x="Marks")`. This creates an interactive histogram showing distribution of marks.	BTL-3
16	**How does Plotly improve data storytelling?** Plotly supports animations, interactivity, and customization that make data insights engaging. It helps convey findings effectively to audiences.	BTL-2
17	**Write a command to create a box plot in Plotly.** Example: `px.box(df, x="Department", y="Salary")`. This shows salary distribution with outliers for each department.	BTL-3
18	**What is the difference between `plotly.express` and `plotly.graph_objects`?**	BTL-2

S.No	Question & Answer	BTL
	`plotly.express` is a high-level interface for quick plots with simple syntax. `plotly.graph_objects` provides detailed customization for complex charts.	
19	**Write a command to create a pie chart in Plotly.** Example: `px.pie(df, names="Category", values="Sales")`. This generates an interactive pie chart with hover labels.	BTL-3
20	**What type of plots can Plotly create?** Plotly supports line, scatter, bar, histogram, box, pie, bubble, and 3D plots. It also provides maps and dashboard visualizations.	BTL-1
21	**What is the purpose of `title` parameter in Plotly charts?** The `title` parameter adds a heading to the chart. Example: `px.line(df, x="Year", y="Sales", title="Sales Over Time")`.	BTL-1
22	**Write a command to save a Plotly chart as HTML.** Example: `fig.write_html("plot.html")`. This saves the chart as an interactive HTML file for web use.	BTL-3
23	**Why is Plotly preferred for big data visualization?** Plotly handles large datasets efficiently with zoom and filter options. It provides smooth performance for interactive analysis.	BTL-2
24	**What is the role of color in Plotly charts?** Colors help differentiate categories in Plotly charts. Example: `color="Gender"` assigns colors to male and female groups.	BTL-2
25	**Write the general syntax of Plotly Express function.** The syntax is: `px.<chart_type>(data_frame, x=column_x, y=column_y, color=column_category, title="chart title")`. The chart type can be line, scatter, bar, etc.	BTL-1

PART B

S.No	Question	BTL
1	Explain the features of Plotly library and demonstrate with a Python program to create a simple line plot.	BTL-2
2	Write a Python program using Plotly Express to create an interactive scatter plot for a dataset with hover information enabled.	BTL-3
3	Differentiate between line plots, bar plots, and scatter plots in Plotly with suitable examples.	BTL-2
4	Write a Python program to visualize sales data using an interactive bar chart with customized colors.	BTL-3
5	Describe hover effects, zooming, and panning features in Plotly with suitable examples.	BTL-2
6	Write a Python program to create a histogram using Plotly and explain how interactive features help in better data analysis.	BTL-3
7	Create a box plot using Plotly Express to analyze salary distribution across different departments and explain the observations.	BTL-4
8	Explain how Plotly can be used for big data visualization. Write a program to demonstrate handling of large dataset using interactive charts.	BTL-4
9	Write a Python program using Plotly to create a pie chart showing product sales distribution. Explain the interactivity features used.	BTL-3
10	Discuss the advantages of using Plotly for interactive visualization compared to Matplotlib and Seaborn. Support your answer with an example program.	BTL-5